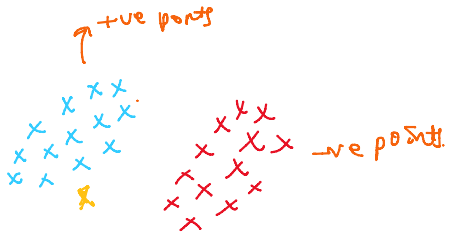
KNN

26 June 2021 10:46

KNN

K – Nearest Neighbours

K → No. of nearest neighbours.
It is a hyper parameter.



X → given test point.

given a test point, it calculates distances of test point from each point
And class label is decided based on nearest neighbours.

Assumption of KNN

Similar points are near to each other, (or)
Nearest points are similar to each other,

If one want to predict the nature of a person, then we could decide by considering his neighbours (family members) nature. (Typically)

This is the main idea of KNN

How many neighbours should we consider is a hyper parameter(k).

ALGORITHM :– [There is no training phase in KNN]

step 1 :– Find the k – nearest neighbour to x_q in D. (dataset)

step 2 :– Decide the class label of x_q based on majority vote.

Say k = 5

$x_1 \rightarrow y_1$

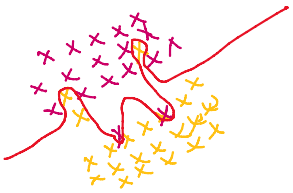
$x_1 \rightarrow y_1$
$x_2 \rightarrow y_2$
$x_3 \rightarrow y_3$
$x_4 \rightarrow y_4$
$x_5 \rightarrow y_5$

} Consider the majority & assign to x_q.

If Regression, then, take mean / median value

K → hyperparameter.

Let us see how KNN behaves for different values of k.



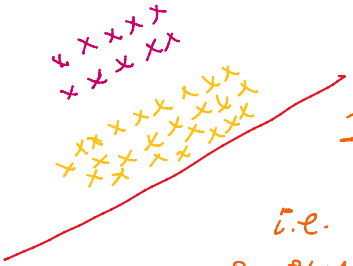
If $K = 1$

If $k=1$, then there is 1-nearest neighbour

So there is overfitting.

i.e. high variance.

⇒ model is sensible to noise.



$k = n$

If $k = n$, then all points are considered as NN.

i.e. class label which is more in dataset will get priority. So every query point will be classified as majority class label in the data set.

So here, there is underfitting

i.e. high bias.

⇒ Model will fail to learn / get any info from the dataset.

... will become dumb.

the dataset.

model will become dumb.

So we have to choose best value for K.

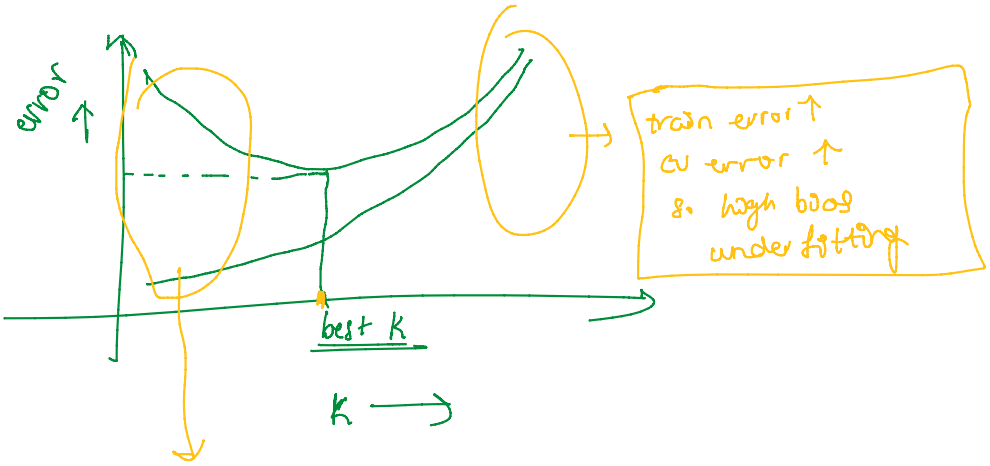
Bias-variance trade off.

So inorder to find best K, we can do cross validation.

D_{train} → NN

D_{CV} → k-value

For better performance we can do k'-fold CV.



train error ↑
CV error ↑
∴ high bias
underfitting

high CV error
low train error
high variance.
Overfitting

for x_q in D_{test} :

pt → x_q

NN → D_{train}
K → D_{CV}

calculate distance of x_q from all points

Calculate distance of x_q from all points.
pick top k close distances & corresponding points.
label using majority vote.

count = 0
if $\hat{y}_q == y_q$:
    count += 1

$$\text{Accuracy} = \frac{\text{count}}{\#\text{pts in test data}}$$

Till now, we gave equal importance to all k-points.

But points which are very close to x_q could have more effect than other NN.

So inorder to assign weightage, we can write like,

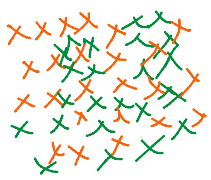
$$w_i = \frac{1}{d_i}$$

Here we are giving more importance to closest NN.

Failure/Limitations of KNN :—

query point is very far away from Dtrain.

K-NN fails here.



when +ve & -ve points are randomly spread over the space.

---

there is no training phase.

Each time when we get x_q, we have to calculate distances from all the points to x_q - So, D_{train} have to be present on RAM to run the models.

Hence space & time complexities are high.

If there is low latency requirement, KNN can't be used.
